Análisis de Clientes mediante Clustering – *Online Retail Dataset*

Contexto general

En el comercio electrónico y el retail, conocer el comportamiento de los clientes es fundamental para diseñar estrategias de marketing efectivas, programas de fidelización y segmentación de mercado.
El **análisis de clustering (agrupamiento)** permite dividir a los clientes en grupos homogéneos según sus características de compra, identificando patrones de consumo y comportamientos similares.

El dataset utilizado en este proyecto proviene del archivo `Online Retail.xlsx`, que contiene datos reales de transacciones de una tienda online.
Incluye información como:

- Identificadores de factura y producto.
- Cantidades vendidas y precios unitarios.
- Fecha de compra y país de origen del cliente.
- Identificador único de cliente.

Objetivo del proyecto

El propósito de este trabajo es **aplicar técnicas de aprendizaje no supervisado** para segmentar a los clientes de acuerdo con su comportamiento de compra.

A través del uso de **métodos de clustering**, se busca:

1. Identificar **grupos de clientes con características de compra similares**.
2. Analizar la contribución económica de cada grupo (valor y frecuencia de compra).
3. Proporcionar información útil para la **toma de decisiones comerciales** y estrategias de marketing personalizadas.

Enfoque metodológico

El análisis se estructura en las siguientes etapas principales:

1. **Carga y exploración de los datos:** comprensión de la estructura del dataset, tipos de variables y limpieza de valores inconsistentes.
2. **Preprocesamiento:** eliminación de duplicados, manejo de valores nulos y creación de métricas relevantes por cliente (por ejemplo, *Recency, Frequency, Monetary* — RFM).

3. **Normalización de variables:** escalado de los datos para evitar sesgos en el agrupamiento.
4. **Aplicación de algoritmos de clustering:**
   - *K-Means Clustering* como técnica principal.
   - Posibles comparaciones con otros métodos (*Hierarchical Clustering* o *DBSCAN*).
5. **Evaluación y visualización de resultados:** análisis de los grupos formados mediante métricas como el **índice de Silhouette** y gráficos interpretativos (2D/3D).
6. **Interpretación de clústeres:** descripción de los perfiles de cliente resultantes (por ejemplo: clientes frecuentes, compradores premium, clientes de bajo valor, etc.).

Resultado esperado

Al finalizar el análisis se obtendrán **segmentos de clientes claramente diferenciados**, que permitirán a la empresa:

- Diseñar campañas personalizadas.
- Priorizar esfuerzos comerciales.
- Comprender mejor el comportamiento de su base de clientes.

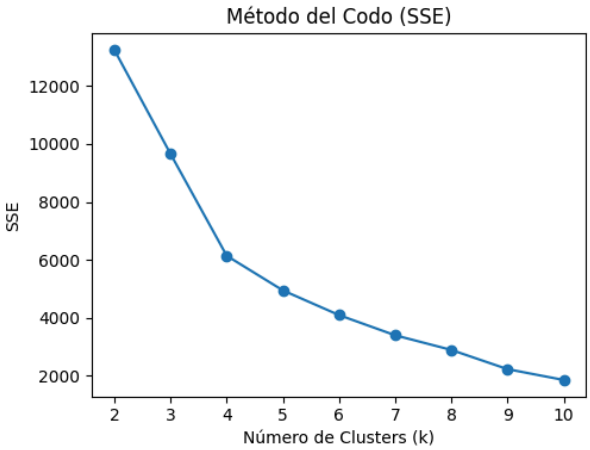
Este tipo de segmentación constituye una base sólida para estrategias de **CRM (Customer Relationship Management)** y marketing basado en datos.

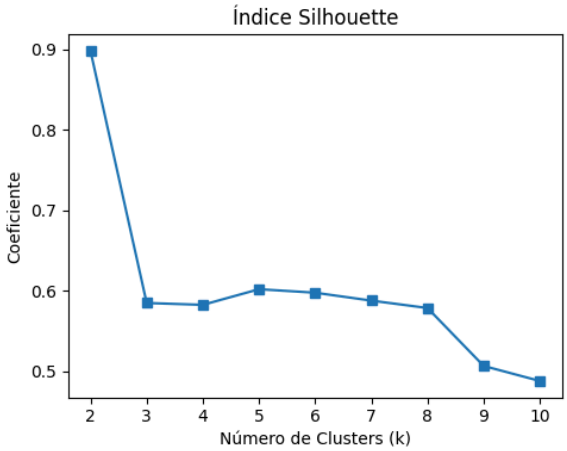
> **Nota:** Este proyecto utiliza técnicas de *Machine Learning no supervisado* enfocadas en descubrimiento de patrones, no en predicción.
> Las conclusiones deben interpretarse en función del contexto del negocio y la naturaleza del dataset.

In [4]:
```
# ========================================
# CLUSTERING — Online Retail.xlsx (plantilla según PPT del profesor)
# ========================================
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score

# ========================================
# 1) CARGA Y SELECCIÓN DE VARIABLES
# ========================================
# Lee el archivo Excel (hoja principal)
df = pd.read_excel("Online Retail.xlsx")

# Limpieza básica (opcional)
df = df.dropna(subset=["CustomerID", "Quantity", "UnitPrice"])
df = df[df["Quantity"] > 0]

# Crear una tabla resumen por cliente
clientes = (
    df.groupby("CustomerID")
```

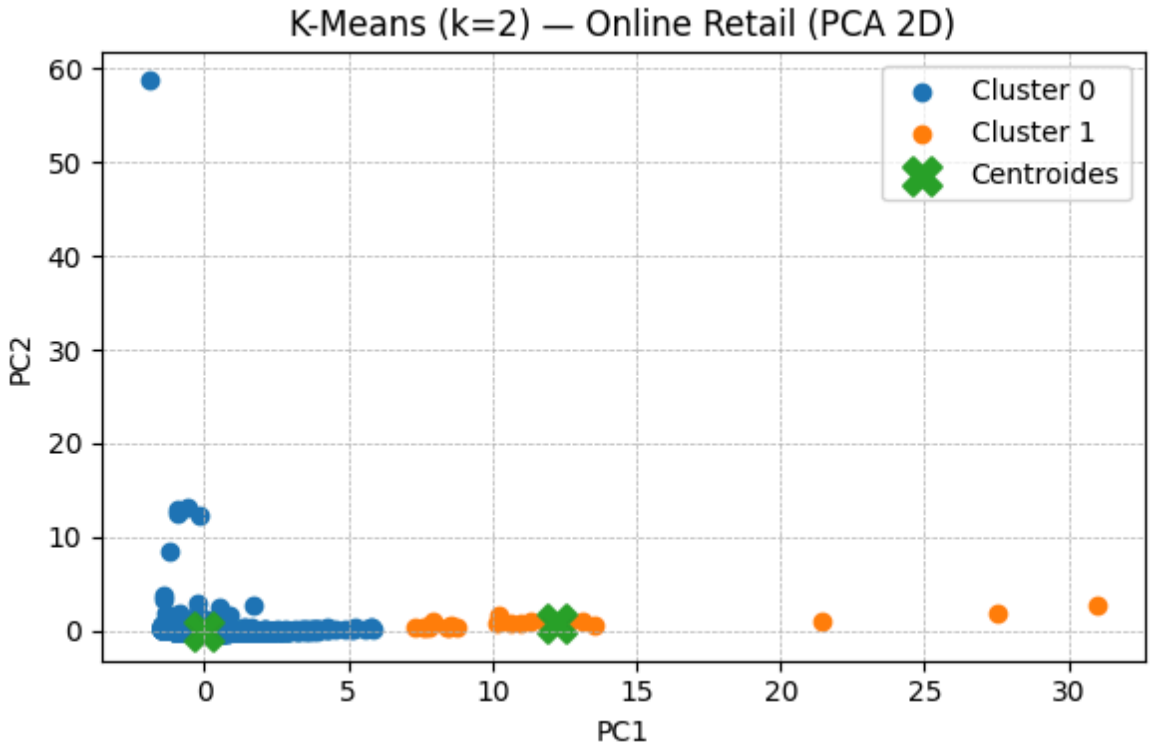
```
    .agg({
        "InvoiceNo": "nunique",       # número de compras
        "Quantity": "sum",            # cantidad total comprada
        "UnitPrice": "mean",          # precio promedio
        "InvoiceDate": "max"          # última fecha de compra
    })
    .rename(columns={
        "InvoiceNo": "Compras",
        "Quantity": "CantidadTotal",
        "UnitPrice": "PrecioPromedio",
        "InvoiceDate": "UltimaCompra"
    })
)

# Calcular variable "Recencia" (días desde última compra)
from datetime import datetime
fecha_ref = clientes["UltimaCompra"].max()
clientes["Recencia"] = (fecha_ref - clientes["UltimaCompra"]).dt.days
clientes = clientes.drop(columns=["UltimaCompra"])

# Variables para clustering (RFM simplificado)
features = ["Recencia", "Compras", "CantidadTotal", "PrecioPromedio"]
X = clientes[features].to_numpy()

# ========================================
# 2) ESCALADO
# ========================================
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# ========================================
# 3) MÉTODO DEL CODO Y SILHOUETTE
# ========================================
max_k = 10
ks = range(2, max_k + 1)
sse = []
sils = []

for k in ks:
    model = KMeans(n_clusters=k, random_state=42, n_init=10)
    labels = model.fit_predict(X_scaled)
    sse.append(model.inertia_)
    sils.append(silhouette_score(X_scaled, labels))

plt.figure(figsize=(10,4))
plt.subplot(1,2,1)
plt.plot(ks, sse, "o-")
plt.title("Método del Codo (SSE)")
plt.xlabel("Número de Clusters (k)")
plt.ylabel("SSE")

plt.subplot(1,2,2)
plt.plot(ks, sils, "s-")
plt.title("Índice Silhouette")
plt.xlabel("Número de Clusters (k)")
plt.ylabel("Coeficiente")
plt.tight_layout()
plt.show()

# ========================================
```

```
# 4) SELECCIÓN AUTOMÁTICA
# ========================================
best_k = ks[np.argmax(sils)]
print(f"Mejor k sugerido (por Silhouette): {best_k}")

# ========================================
# 5) MODELO FINAL
# ========================================
final_kmeans = KMeans(n_clusters=best_k, random_state=42, n_init=10)
labels = final_kmeans.fit_predict(X_scaled)
clientes["Cluster"] = labels
centroids = scaler.inverse_transform(final_kmeans.cluster_centers_)

# ========================================
# 6) REPORTE TIPO WEKA
# ========================================
print(f"Silhouette (k={best_k}): {silhouette_score(X_scaled, labels):.3f}")
print("Tamaño por clúster:", clientes["Cluster"].value_counts().sort_index().to_

header = "Variable".ljust(15) + "".join([f" {i:>8}" for i in range(len(centroids
print("Cluster centroids:")
print(header)
print("-" * len(header))
for j, var in enumerate(features):
    row = f"{var:<15}" + "".join([f"{centroids[i][j]:>9.1f}" for i in range(len(
    print(row)

# ========================================
# 7) GRÁFICO (PCA si >2 variables)
# ========================================
from sklearn.decomposition import PCA

pca = PCA(n_components=2, random_state=42)
X_pca = pca.fit_transform(X_scaled)
C_pca = pca.transform(final_kmeans.cluster_centers_)

plt.figure(figsize=(6,4))
for lab in np.unique(labels):
    m = labels == lab
    plt.scatter(X_pca[m, 0], X_pca[m, 1], label=f"Cluster {lab}")

plt.scatter(C_pca[:,0], C_pca[:,1], marker="X", s=180, linewidths=1.5, label="Ce
plt.title(f"K-Means (k={best_k}) — Online Retail (PCA 2D)")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.legend()
plt.grid(True, linestyle="--", linewidth=0.5)
plt.tight_layout()
plt.show()

# ========================================
# 8) GUARDAR RESULTADOS
# ========================================
clientes.to_csv("OnlineRetail_clusters.csv", index=True)
print("\n✅ Archivo generado: OnlineRetail_clusters.csv")
```






```
Mejor k sugerido (por Silhouette): 2
Silhouette (k=2): 0.898
Tamaño por clúster: {0: 4315, 1: 24}

Cluster centroids:
Variable               0        1
---------------------------------
Recencia            91.9     18.5
Compras              3.9     68.3
CantidadTotal      912.0  51937.6
PrecioPromedio       4.5      4.0
```



✅ Archivo generado: OnlineRetail_clusters.csv